

# ASSIGNMENT-3.3

2303A51295

BATCH-10

## **TASK-1:**

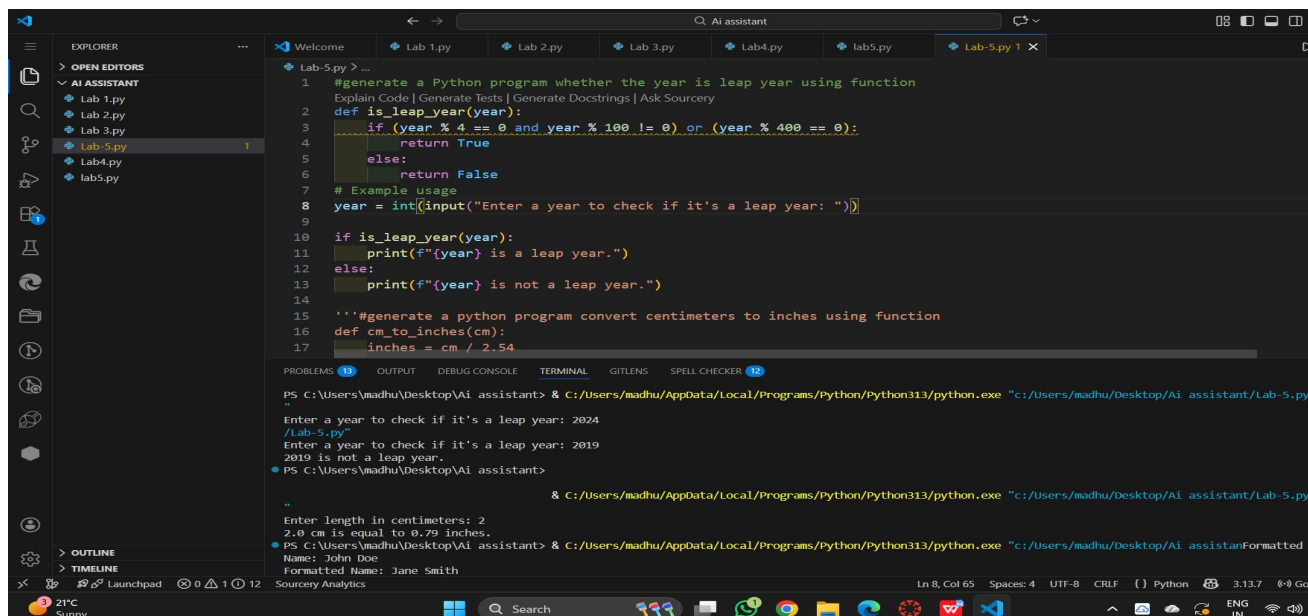
### **PROMPT:**

generate a Python program whether the year is leap year using function

### **CODE:**

```
def is_leap_year(year):  
    if (year % 4 == 0 and year % 100 != 0) or (year %  
400 == 0):  
        return True  
    else:  
        return False  
  
# Example usage  
year = int(input("Enter a year to check if it's a leap  
year: "))  
  
if is_leap_year(year):  
    print(f"{year} is a leap year.")  
else:  
    print(f"{year} is not a leap year.")
```

### **OUTPUT:**



The screenshot shows a Visual Studio Code editor with a Python file named 'Lab-5.py'. The code defines a function 'is\_leap\_year' and includes example usage. The terminal window at the bottom shows the execution of the program. It prompts the user to enter a year, and the user enters 2024, which is correctly identified as a leap year. The user then enters 2019, which is correctly identified as not a leap year. The terminal also shows the execution of a command to format the code using 'black'.

```
PS C:\Users\madhu\Desktop\AI assistant> & C:\Users\madhu\AppData\Local\Programs\Python\Python313\python.exe "c:/Users/madhu/Desktop/AI assistant/Lab-5.py"  
Enter a year to check if it's a leap year: 2024  
2024 is a leap year.  
Enter a year to check if it's a leap year: 2019  
2019 is not a leap year.  
PS C:\Users\madhu\Desktop\AI assistant>  
PS C:\Users\madhu\Desktop\AI assistant> & C:\Users\madhu\AppData\Local\Programs\Python\Python313\python.exe "c:/Users/madhu/Desktop/AI assistant/Lab-5.py"  
Enter length in centimeters: 2  
2.0 cm is equal to 0.79 inches.  
PS C:\Users\madhu\Desktop\AI assistant> & C:\Users\madhu\AppData\Local\Programs\Python\Python313\python.exe "c:/Users/madhu/Desktop/AI assistant/Lab-5.py"  
Formatted Name: Jane Smith  
Formatted Name: Jane Smith
```

### **EXPLANATION:**

- This program determines whether a given year is a leap year using a function.

- The function applies standard leap year rules and returns True or False.
- The user inputs a year, which is checked by the function
- The result is printed as either a leap year or not.

## **TASK-2**

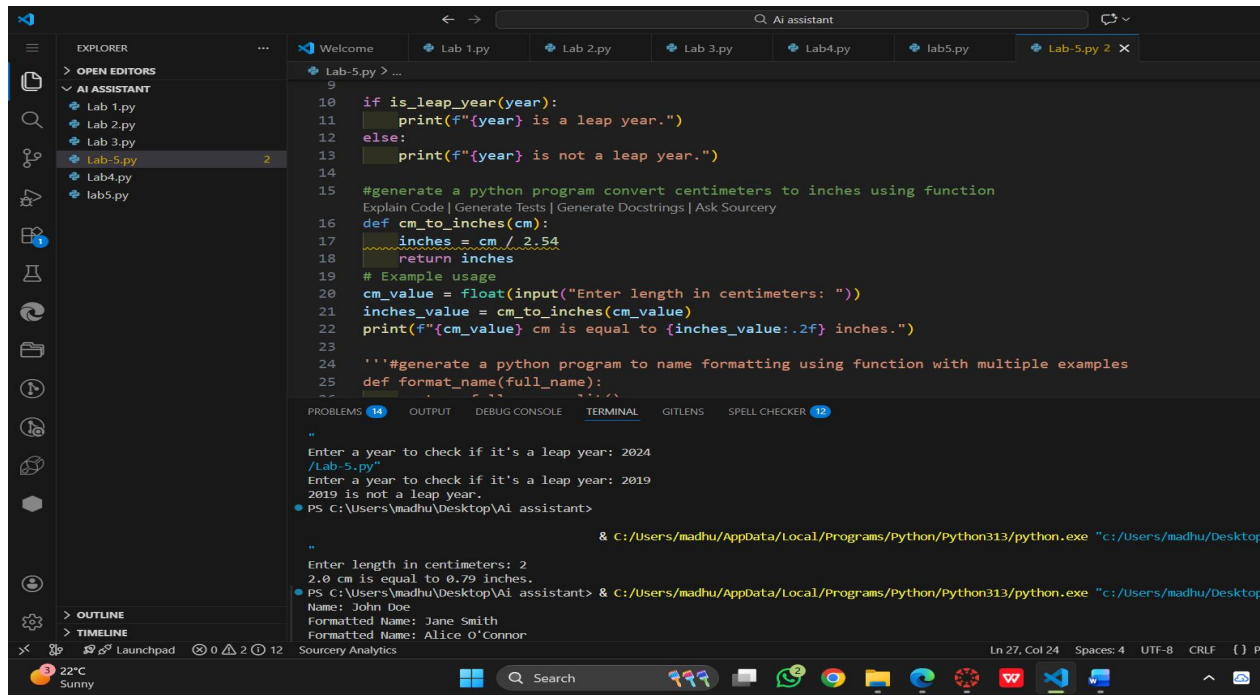
### **PROMPT:**

generate a python program convert centimeters to inches using function

### **CODE:**

```
def cm_to_inches(cm):  
    inches = cm / 2.54  
    return inches  
# Example usage  
cm_value = float(input("Enter length in centimeters: "))  
inches_value = cm_to_inches(cm_value)  
print(f"{cm_value} cm is equal to {inches_value:.2f} inches.")
```

### **OUTPUT:**



### EXPLANATION:

- This program converts a length from centimeters to inches using the correct mathematical formula.
- A function performs the conversion by dividing the value by 2.54.
- The user enters a value in centimeters, which is passed to the function.
- The converted result is displayed in inches.

### TASK-3:

#### PROMPT:

#generate a python program to name formatting using function with multiple examples

#### CODE:

```

def format_name(full_name):
    parts = full_name.split()
    if len(parts) >= 2:
        first_name = parts[0]
        last_name = parts[-1]
        return f"{last_name}, {first_name}"
    else:
        return full_name # Return as is if not enough parts

# Example usage
test_names = [
    "John Smith",
    "Anita Rao",
    "Alice Johnson",
    "Bob"

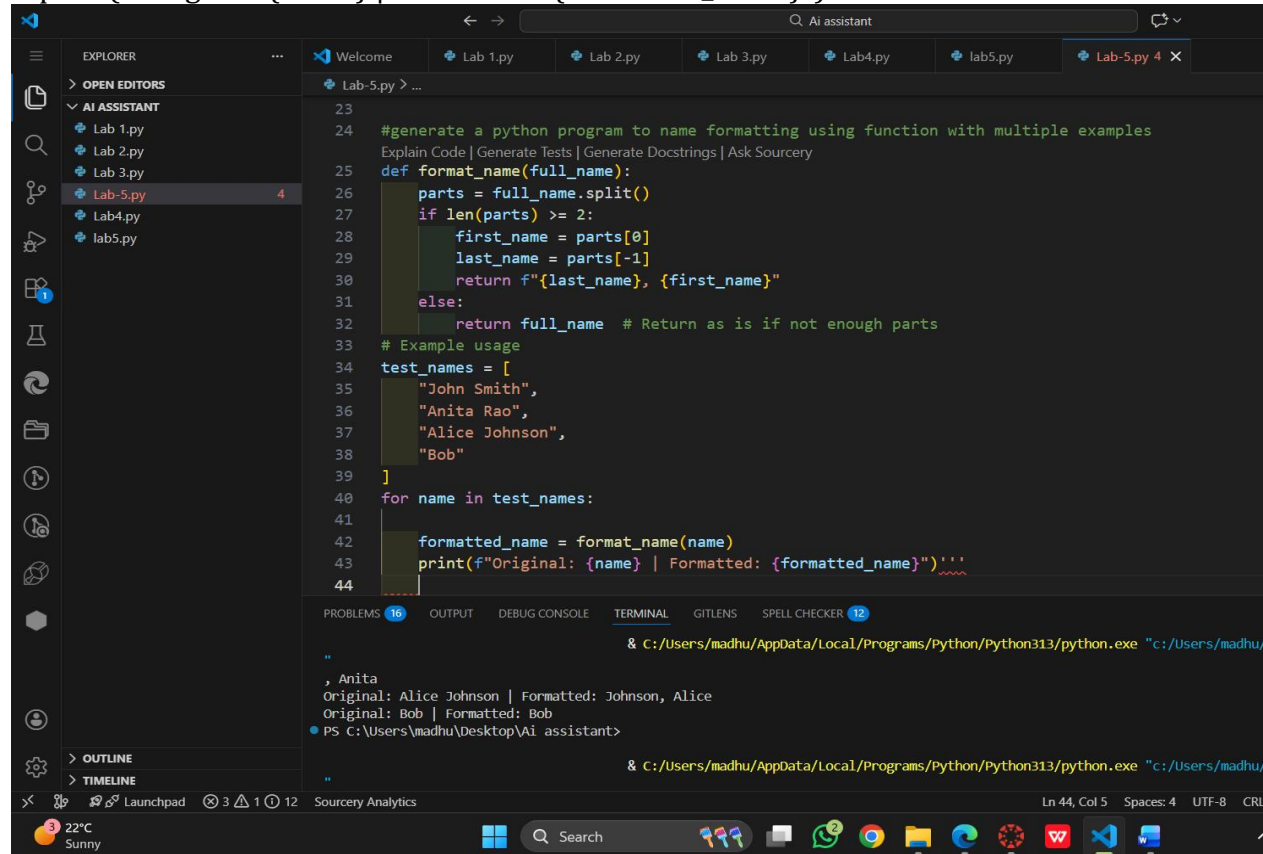
```

]

for name in test\_names:

    formatted\_name = format\_name(name)

    print(f"Original: {name} | Formatted: {formatted\_name}")



The screenshot shows a Visual Studio Code editor with a Python file named 'Lab-5.py'. The code defines a function 'format\_name' that splits a full name into first and last names and formats them. It then uses this function to format a list of names: 'John Smith', 'Anita Rao', 'Alice Johnson', and 'Bob'. The terminal output shows the results of the formatting process.

```
23
24 #generate a python program to name formatting using function with multiple examples
25 Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
26 def format_name(full_name):
27     parts = full_name.split()
28     if len(parts) >= 2:
29         first_name = parts[0]
30         last_name = parts[-1]
31         return f"{last_name}, {first_name}"
32     else:
33         return full_name # Return as is if not enough parts
34 # Example usage
35 test_names = [
36     "John Smith",
37     "Anita Rao",
38     "Alice Johnson",
39     "Bob"
40 ]
41 for name in test_names:
42     formatted_name = format_name(name)
43     print(f"Original: {name} | Formatted: {formatted_name}")
44
```

Terminal Output:

```
& C:/Users/madhu/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/madhu/
, Anita
Original: Alice Johnson | Formatted: Johnson, Alice
Original: Bob | Formatted: Bob
PS C:\Users\madhu\Desktop\Ai assistant>
```

**OUTPUT:**

**EXPLANATION:**

- This program formats a full name entered as first name and last name.
- The input is split and validated to ensure the correct format.
- Each part of the name is cleaned and capitalized.
- The formatted full name is then displayed.

## TASK-4:

**PROMPT:**

generate a python code using function that counts vowels in a string

**CODE:**

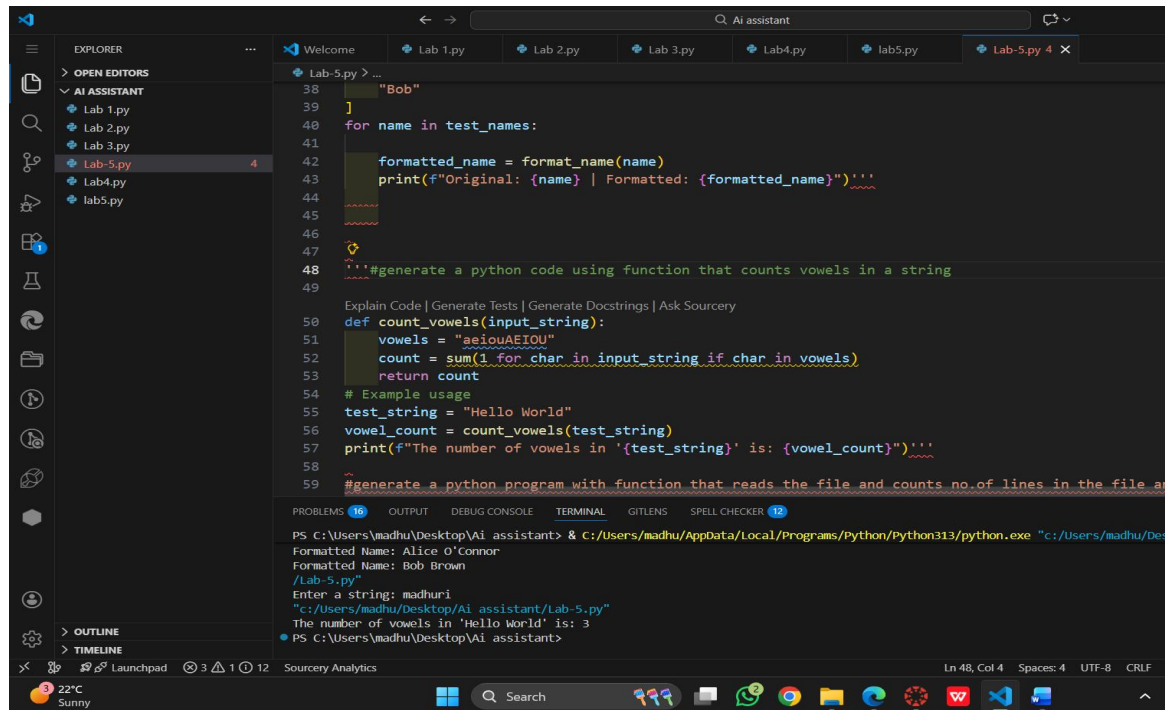
```
def count_vowels(input_string):
```

```
    vowels = "aeiouAEIOU"
```

```
    count = sum(1 for char in input_string if char in vowels)
```

```
    return count
# Example usage
test_string = "Hello World"
vowel_count = count_vowels(test_string)
print(f"The number of vowels in '{test_string}' is: {vowel_count}")
```

**OUTPUT:**



## EXPLANATION:

- The function counts vowels in a given string using a direct logic approach.
- Zero-shot prompting applies the logic without examples.
- Few-shot prompting helps by showing patterns before execution.
- The function returns the total number of vowels in the input string.

## TASK-5:

### PROMPT:

generate a python program with function that reads the file and counts no. of lines in the file and return the line count

### CODE:

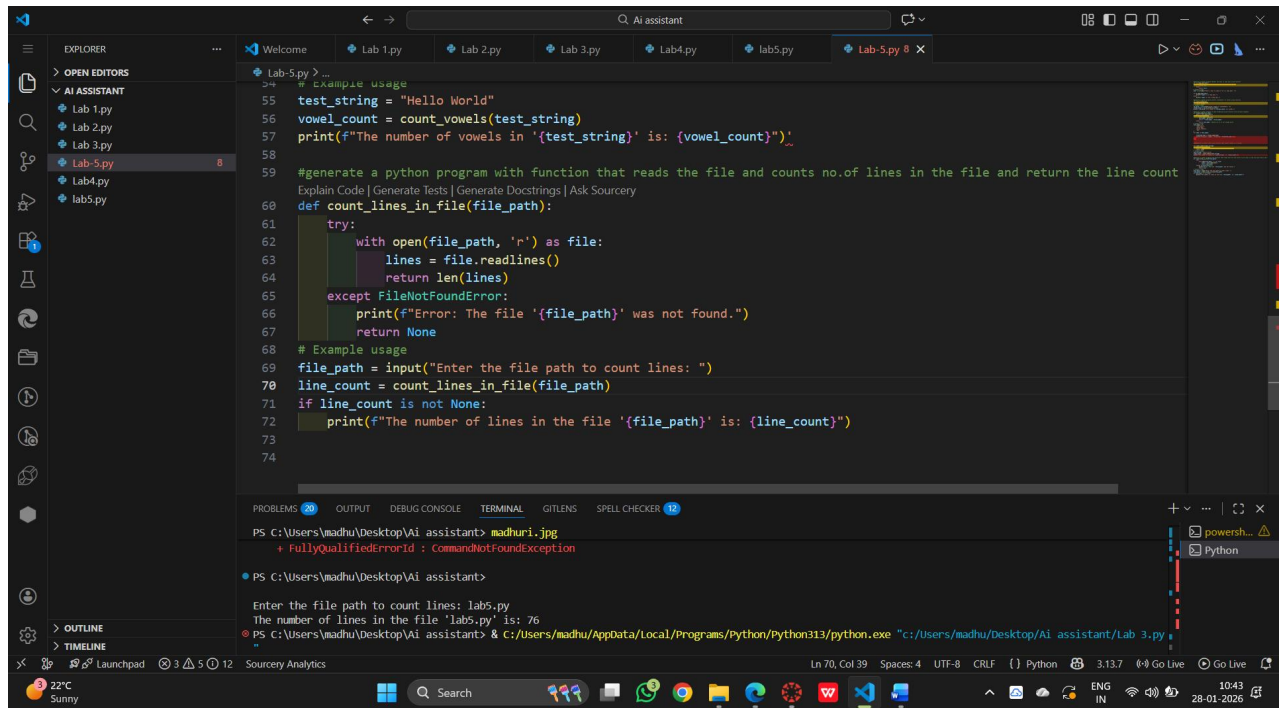
```
def count_lines_in_file(file_path):
    try:
        with open(file_path, 'r') as file:
            lines = file.readlines()
            return len(lines)
    except FileNotFoundError:
        print(f'Error: The file '{file_path}' was not found.')
        return None

# Example usage
file_path = input("Enter the file path to count lines: ")
line_count = count_lines_in_file(file_path)
```

if line\_count is not None:

```
print(f"The number of lines in the file '{file_path}' is: {line_count}")
```

## OUTPUT:



The screenshot shows a Visual Studio Code editor with a Python file named 'Lab 5.py' open. The code defines a function 'count\_lines\_in\_file' that reads a file and returns the number of lines. It includes an example usage section where it prompts the user for a file path and prints the line count. The terminal at the bottom shows the execution of the script, which successfully counts 76 lines in the file 'lab5.py'.

```
54 # Example usage
55 test_string = "Hello World"
56 vowel_count = count_vowels(test_string)
57 print(f"The number of vowels in '{test_string}' is: {vowel_count}")
58
59 #generate a python program with function that reads the file and counts no.of lines in the file and return the line count
60 def count_lines_in_file(file_path):
61     try:
62         with open(file_path, 'r') as file:
63             lines = file.readlines()
64             return len(lines)
65     except FileNotFoundError:
66         print(f"Error: The file '{file_path}' was not found.")
67         return None
68
69 # Example usage
70 file_path = input("Enter the file path to count lines: ")
71 line_count = count_lines_in_file(file_path)
72 if line_count is not None:
73     print(f"The number of lines in the file '{file_path}' is: {line_count}")
74
```

Terminal Output:

```
PS C:\Users\madhu\Desktop\Ai assistant> madhuri.jpg
+ FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\madhu\Desktop\Ai assistant>
Enter the file path to count lines: lab5.py
The number of lines in the file 'lab5.py' is: 76
PS C:\Users\madhu\Desktop\Ai assistant> & c:/Users/madhu/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/madhu/Desktop/Ai assistant/Lab 3.py"
```

**EXPLANATION:**

- This program reads a text file and counts the number of lines.
- A function opens the file safely and calculates the line count.
- Error handling is used if the file does not exist.
- The final line count is returned and displayed.